CSE 490V Final Project Proposal

MICHAL PISZCZEK

1 ELEVATOR PITCH

I will combine VR with ray-tracing. My project will aim to recreate our final deliverable from Homework 4 (a scene properly rendered in stereo such that it appears 3D through our headsets) but with ray-tracing. This means finding or implementing a simple ray-tracer, and then implementing the algorithm by Stephen et al. [1996] to be able to efficiently generate approximate ray-traced stereo images. Key challenges will be finding or building a simple ray-tracer that can be hooked into our existing setup, and then implementing the algorithm described in that paper. The final deliverable will feature a scene with an eye-chart, rather than the standard teapot, to really show off the detail captured by ray-tracing.

2 EXTENDED OVERVIEW

Ray-tracing in real-time is already a computationally taxing endeavor, and adding stereo to the mix further adds to the load. To effectively render in stereo, the naive approach is to simply render a separate image for each eye, as we have done in class. When producing standard renderings, we can accept this additional cost and power through, but with ray-tracing, each frame already takes so long to render, that having to do so twice significantly degrades performance. There has been some success at tackling this problem in industry and academia, including special hardware support from GPU makers like Nvidia [2019] and algorithms that take advantage of this, like one recently developer by Silva et al. [2019]. I will be attempting another approach, which is to approximately render one of the two required images per frame, using the algorithm developed by Stephen et al. [1996]. This algorithm claims to reduce the computational cost of creating the right-eye image by over 80 percent when already given the ray-traced left-eye image, at an acceptable level of quality loss. Hopefully this will allow for a decently immersive ray-traced VR experience, which I will attempt to demo with a simple scene featuring an eye-chart.

2.1 Technical Challenges

- Finding or implementing a simple ray-tracer that can fit into our existing pipeline.
- Implementing the algorithm developed by Stephen et al. [1996].
- Replacing the standard teapots with an eye-chart in the demo scene.

2.2 Key Risks and Mitigation

- A potential risk is that I may not be able to find or quickly implement a simple ray-tracer that fits into our rendering pipeline. If this starts seeming like it will be the case, I have two viable back-up plans: 1) commit to the ray-tracing and make the focus of the project about me learning how to implement a ray-tracer 2) forget about the ray-tracer, and see if I can adapt the algorithm I found (or use another) to speed up our existing stereo rendering system by generating one of the two images approximately.
- Another risk is the algorithm I found not being described well enough for me to be able to implement it, or that I simply can't within the time-frame. Then, as with the first risk, the project can simply fall back to me learning about ray-tracers, without the efficient

Author's address: Michal Piszczek, michalp@cs.washington.edu.

stereo rendering component. Alternatively, I could also attempt an efficient foveation-based approach to ray-traced stereo-rendering like the one given by Fujita et al. [2014].
- The final risk is that I am unable to find and place (in-scene) a suitable eye-chart for my demo. If so, I will just keep the teapots.

3 HARDWARE AND SOFTWARE

I will only require the software/hardware that I used to complete Homework 4. I can't think of a way to incorporate a professional headset into my project. Though, if you have an idea of how I can, or any extra Occuli/Hololenses/MagicLeaps/Indices, then I'd be more than happy to borrow one for a little! :)

4 TEAM RESPONSIBILITIES

- **Student One**: Responsible for: (1) finding/implementing a ray-tracer and hooking it into our course headset's pipeline (2) implementing the approximate stereo rendering algorithm (3) swapping out the teapots for an eye-chart.

5 DEVELOPMENT PLAN

- **March 06**: Ray-tracer found or implemented
- **March 12**: Approximate stereo rendering algorithm implemented
- **March 13**: Eye-chart swapped in, teapots swapped out
- **March 17**: Paper done
- **March 18**: Demo done/tested

REFERENCES

Stephen J. Adelson and Larry F. Hodges. 1996. Visible Surface Ray-Tracing of Stereoscopic Images. In *ACM-SE 30: Proceedings of the 30th annual Southeast regional conference*. Association for Computing Machinery, 148–156.

Masahiro Fujita and Takahiro Harada. 2014. Foveated Real-Time Ray Tracing for Virtual Reality Headset.

Vinicius Silva and Luiz Velho. 2019. Immersive Visualization of the Classical Non-Euclidean Spaces using Real-Time Ray Tracing.

David Weinstein. 2019. From Foveated Rendering to Photorealism: See How NVIDIA RTX Powers Deeper VR Immersion. (2019).